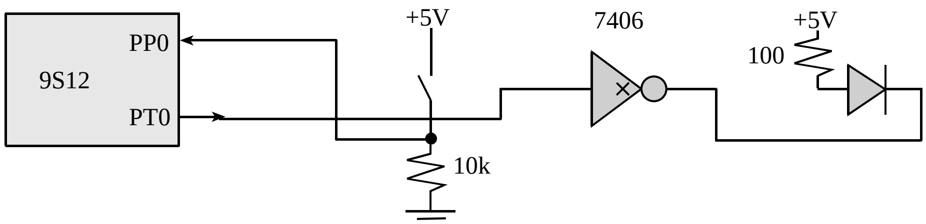
(4) Question 1. The basis elements are 1000=27, 0100=9, 0010=3, and 0001=1* 27+0*9+2*3+1=34
(3) Question 2. Answer true/false for each of the following three statements
Part a) True, the stack pointer (SP) points to the data on top of the stack.
Part b) False, the order in which I add the numbers does not affect the final value of RegA.
Part c) False, dropout error cannot occur on a logical left shift (e.g., `lsla`). Overflow can occur.
(4) Question 3. Consider `ldab #-6  subb #251`
Convert to signed, 251 = 251-256 = -5. Subtract two signed -6 - -5 is -1. This fits so V=0.
Convert to unsigned -6 = -6+256 = 250. Subtract unsigned 250-251 is -1. Does not fit, C=1.
(4) Question 4. What is the binary representation of 8-bit signed number -11?
Method 1) +11 is 8+2+1 or 00001011. Negative is 2's complement. Complement 1111,0100, then add 1. **11110101**
Method 2) Look at basis elements, need -128,64,32,16,4,1, so **11110101**
Method 3) -11 is the same binary as -11+256 = 245. 245/16=15 remainder 5. So hex is $F5
(20) Question 5. The current through LED resistor 25mA = (5-2-0.5)/R. Solve for R= 2.5V/25mA = 100Ω. The pull down resistor on the switch could be 10kΩ or 100kΩ. I will even count 1kΩ or 1MΩ.



(5) Question 6. The bus cycles occurring for `stx $3000`

R/W	Addr	Data	Changes to D,X,Y,S,PC,IR,EAR
R	$4200	$7E	PC=$4201,IR=$7E
R	$4201	$30	PC=$4202
R	$4202	$00	PC=$4203,EAR=$3000
W	$3000	$12	
W	$3001	$34	

(20) Question 7. Mask the bits of interest, then compare.

```
; fastest execution
Check ldaa PTT  ;read all 8 bits
      anda #$45 ;look at just bits 6,2,0
      cmpa #$01 ;expected value
      bne  done
      bset PTT,#$80 ;PT0=1, PT2=0, and PT6=0 so make PT7=1
done  rts
```

```
;simple to understand
Check ldaa PTT  ;read all 8 bits
      bita #$44 ;look at bits 6,2
      bne  done ;skip if either PT6 or PT2 are 1
      bita #$01 ;look at bit 0
      beq  done ;skip if PT0 is 0
      oraa #$80 ;PT0=1, PT2=0, and PT6=0 so make PT7=1
      staa PTT
done  rts
```

```
;fewest number of instructions
Check brset PTT,#$44,done ;skip if either PT6 or PT2 are 1
      brclr PTT,#$01,done ;skip if PT0 is 0
      bset  PTT,#$80 ;PT0=1, PT2=0, and PT6=0 so make PT7=1
done  rts
```

(20) Question 8. Write an assembly language subroutine that adds two unsigned 16-bit numbers.

```
;simple to understand
      org  $2000 ;RAM
yval  rmb  2
      org  $4000
add   sty  yval   ;save in variable
      tfr  x,d
      addd yval   ;add two inputs
      bcc  ok
      ldd  #65535 ;ceiling on overflow
ok    rts
```

```
;uses stack, so no global is required
add   pshy        ;save Y on stack
      tfr  x,d
      addd 2,sp+  ;add two inputs
      bcc  ok
      ldd  #65535 ;ceiling on overflow
ok    rts
```

---

(20) Question 9. A subroutine that counts the number of binary bits that are zero.

```
;simple to understand
Count clrb      ;result
      ldx  #8   ;loop counter
loop  lsra      ;bit into carry (could shift right or left)
      bcs  skip
      incb      ;found a zero
skip  dbne x,loop
      rts
```

```
;fastest to execute, does not require a loop counter
Count clrb      ;result
      coma      ;will be counting 1's now
loop  bpl  skip ;bit7=0, do not count
      incb      ;found a 1 (means found a 0)
      lsla      ;move bits into bit7
      bne  loop ;done when A=0
      rts
```